



FPT POLYTECHNIC



android

www.poly.edu.vn

LẬP TRÌNH ANDROID 2

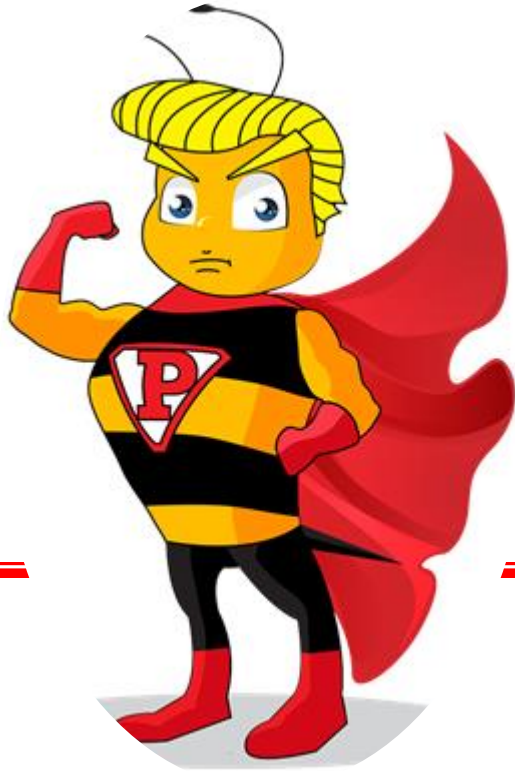
FIREBASE

- ☐ Tổng quan về Firebase
- ☐ Tạo dự án trên Firebase
- ☐ CRUD trong Firebase (Firebase Cloud Firestore)

MỤC TIÊU

- ◎ TỔNG QUAN VỀ FIREBASE
- ◎ TẠO DỰ ÁN TRÊN FIREBASE
- ◎ CRUD TRONG FIREBASE (FIREBASE CLOUD FIRESTORE)





FIREBASE

...

- ❑ **Firestore** là một nền tảng giúp phát triển các ứng dụng di động. Bên cạnh đó, Firestore còn được xem là một dịch vụ cơ sở dữ liệu hoạt động trên nền tảng đám mây cloud với hệ thống máy chủ mạnh mẽ của Google.
- ❑ **Firestore** chứa cơ sở dữ liệu mang đến khả năng code nhanh và thuận tiện hơn. Lập trình viên có thể dễ dàng lập trình ứng dụng bằng cách đơn giản hóa các thao tác với cơ sở dữ liệu sẵn có.




- ❑ Vào năm 2011, James Tamplin và Andrew Lee đã cho ra mắt **Evolve**. Đây là một nền tảng có cấu trúc khá đơn giản chuyên cung cấp các API cần thiết để tích hợp tính năng trò chuyện vào các trang web. Tuy nhiên, họ nhận ra rằng nền tảng này đang được sử dụng để truyền dữ liệu ứng dụng chứ không đơn giản là chat. Sau đó, họ đã phát triển **Envole** và tạo nên **Firestore**.
- ❑ Đến tháng 4 năm 2012, Firestore đã được công bố trên toàn cầu dưới dạng một công ty riêng biệt. Những năm sau đó, **Firestore** đã thực hiện nhiều cuộc huy động vốn và ra mắt các sản phẩm mới.
- ❑ Đến tháng 10 năm 2014, **Firestore** đã chính thức được **Google** mua lại và trở thành một ứng dụng đa năng trên nền tảng di động và website.

❑ Realtime Database

- ❑ Realtime Database là một cơ sở dữ liệu thời gian thực được lưu trữ dưới dạng JSON và được đồng bộ hóa theo thời gian thực đối với mọi kết nối.
- ❑ Đối với các ứng dụng được xây dựng trên đa nền tảng như Android, iOS và WebApp, tất cả client sẽ cùng sử dụng một cơ sở dữ liệu. Bên cạnh đó, hệ thống dữ liệu này sẽ tự động cập nhật khi lập trình viên phát triển ứng dụng. Sau đó, tất cả dữ liệu này sẽ được truyền tải thông qua các kết nối SSL có 2048 bit.




Authentication

-  Authentication là tính năng giúp xác thực danh tính của người dùng ứng dụng. Firebase cung cấp các bước xác thực thông qua Email, Facebook, Twitter, GitHub hay Google. Điều này giúp cho các thông tin cá nhân của khách hàng được bảo vệ một cách tốt nhất, hạn chế được tình trạng bị hacker đánh cắp. Đồng thời việc xác thực danh tính qua Firebase sẽ giúp người dùng tiếp cận sản phẩm nhanh chóng và an toàn hơn.




Cloud Storage

-  Cloud Storage là tính năng cho phép lưu trữ và quản lý nội dung đã tạo ra như ảnh, video, nội dung, văn bản,... Firebase Storage cung cấp các API hỗ trợ bạn upload và download các file từ ứng dụng một cách trơn tru mà không cần quan tâm đến chất lượng đường truyền mạng với độ bảo mật cao.



Cloud Storage
for Firebase

Hosting

-  Hosting được phân phối thông qua tiêu chuẩn công nghệ bảo mật SSL từ hệ thống mạng CDN. CDN là một mạng lưới máy chủ giúp lưu trữ các bản sao của các nội dung tĩnh trên website. Thông qua CDN, người dùng có thể truy cập và sử dụng các dịch vụ trên web khi cài Firebase Hosting một cách nhanh chóng và ổn định hơn.



□ Analytics

- Analytics giúp bạn có thể phân tích hành vi của người sử dụng ứng dụng của bạn. Qua đó, bạn sẽ biết được khách hàng thường xuyên truy cập tính năng nào và các thông tin về hiệu quả quảng cáo, tình trạng trả phí,... để có thể đưa ra được chiến lược phát triển phù hợp. Để thực hiện tính năng Analytics của Firebase, bạn cần cài đặt Software Development Kit (SDK).



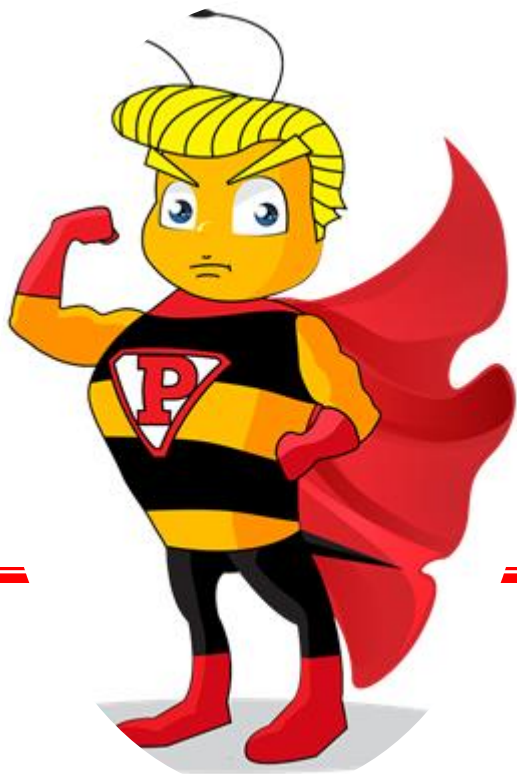
Google Analytics
for Firebase

□ Ưu điểm

- Sử dụng miễn phí và thuận tiện
- Dễ sử dụng và tích hợp
- Đáp ứng nhu cầu của người dùng
- Cập nhật liên tục và đa nền tảng

□ Nhược điểm

- ❖ Giới hạn về quy mô ứng dụng
- ❖ Khả năng tùy chỉnh hạn chế
- ❖ Giá các dịch vụ của Firebase



TẠO DỰ ÁN TRÊN FIREBASE

...





XỬ LÝ MỘT SỐ THAO TÁC TRÊN FIREBASE FIRESTORE

...

❑ Lắng nghe dữ liệu trên FirebaseFirestore khi thêm 1 dòng dữ liệu mới

```
database.collection(TODO).addSnapshotListener(new EventListener<QuerySnapshot>() {  
    @Override  
    public void onEvent(@Nullable QuerySnapshot value, @Nullable FirebaseFirestoreException error) {  
        if (error != null) {  
            Log.e(tag: "TAG", msg: "Listen failed", error);  
            return;  
        }  
  
        if (value != null) {  
            for (DocumentChange dc : value.getDocumentChanges()) {  
                if (dc.getType() == DocumentChange.Type.ADDED) { //Sự kiện khi có 1 document được thêm vào  
                    dc.getDocument().toObject(Todo.class);  
                    listToDo.add(dc.getDocument().toObject(Todo.class));  
                    listToDoAdapter.notifyItemInserted(position: listToDo.size() - 1);  
                }  
            }  
        }  
    }  
});
```


❑ Lắng nghe dữ liệu trên FirebaseFirestore khi thay đổi thông tin 1 dòng dữ liệu

```
database.collection(TODO).addSnapshotListener(new EventListener<QuerySnapshot>() {  
    @Override  
    public void onEvent(@Nullable QuerySnapshot value, @Nullable FirebaseFirestoreException error) {  
        if (error != null) {  
            Log.e( tag: "TAG", msg: "Listen failed", error);  
            return;  
        }  
  
        if (value != null) {  
            for (DocumentChange dc : value.getDocumentChanges()) {  
                if (dc.getType() == DocumentChange.Type.MODIFIED) {  
                    ToDo updateToDo = dc.getDocument().toObject(ToDo.class);  
                    //Nếu vị trí của đối tượng tương ứng động với vị trí mới  
                    if (dc.getOldIndex() == dc.getNewIndex()) {  
                        listToDo.set(dc.getOldIndex(), updateToDo);  
                        listToDoAdapter.notifyItemChanged(dc.getOldIndex());  
                    } else {  
                        listToDo.remove(dc.getOldIndex());  
                        listToDo.add(updateToDo);  
                        listToDoAdapter.notifyItemMoved(dc.getOldIndex(), dc.getNewIndex());  
                    }  
                }  
                //Trường hợp xảy ra các điều kiện trên là do bản thân của firebase  
                // sẽ sắp xếp lại dữ liệu cho chúng ta (nếu chúng ta ko đặc điều kiện sắp xếp cụ thể khi truy vấn)  
            }  
        }  
    }  
});
```

❑ Lắng nghe dữ liệu trên FirebaseFirestore khi xóa 1 dòng dữ liệu

```
database.collection(TODO).addSnapshotListener(new EventListener<QuerySnapshot>() {  
    @Override  
    public void onEvent(@Nullable QuerySnapshot value, @Nullable FirebaseFirestoreException error) {  
        if (error != null) {  
            Log.e(tag: "TAG", msg: "Listen failed", error);  
            return;  
        }  
  
        if (value != null) {  
            for (DocumentChange dc : value.getDocumentChanges()) {  
                if (dc.getType() == DocumentChange.Type.REMOVED) {  
                    dc.getDocument().toObject(ToDo.class);  
                    listToDo.remove(dc.getOldIndex());  
                    listToDoAdapter.notifyItemRemoved(dc.getOldIndex());  
                }  
            }  
        }  
    }  
});
```

❑ Thêm một dữ liệu mới

```
database.collection(TODO).document(id) // Dùng id để đặt tên cho document
    .set(mapTodo) // Đưa dữ liệu vào database
    .addOnSuccessListener(new OnSuccessListener<Void>() {
        @Override
        public void onSuccess(Void unused) {
            Toast.makeText(context: MainActivity.this, text: "Thêm công việc thành công", Toast.LENGTH_SHORT).show();
        }
    }).addOnFailureListener(new OnFailureListener() {
        @Override
        public void onFailure(@NonNull Exception e) {
            Toast.makeText(context: MainActivity.this, text: "Thất bại", Toast.LENGTH_SHORT).show();
        }
    });
```

❑ Thay đổi thông tin của một dữ liệu

```
database.collection(TODO).document(todo.getId()) //gán ID Document cần update
    .update(todo.convertHashMap())
    .addOnSuccessListener(new OnSuccessListener() {
        @Override
        public void onSuccess(Object o) {
            Toast.makeText(context: MainActivity.this, text: "Cập nhật thành công", Toast.LENGTH_SHORT).show();
        }
    }).addOnFailureListener(new OnFailureListener() {
        @Override
        public void onFailure(@NonNull Exception e) {
            Log.e(tag: "Lỗi", msg: "onFailure: " + e);
        }
    });
```

❑ Xóa thông tin của một dữ liệu

```
database.collection(TODO).document(id) //truyền id công việc cần xóa
.delete().addOnSuccessListener(new OnSuccessListener() {
    @Override
    public void onSuccess(Object o) {
        Toast.makeText(context: MainActivity.this, text: "Xóa thành công", Toast.LENGTH_SHORT).show();
    }
}).addOnFailureListener(new OnFailureListener() {
    @Override
    public void onFailure(@NonNull Exception e) {
        Log.e(tag: "Lỗi", msg: "onFailure: " + e);
    }
});
```

DEMO



FPT POLYTECHNIC

Thank you